UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/692,320	10/23/2003	Krzysztof J. Cwalina	MS1-1748US	8610

22801 7590 04/11/2007
LEE & HAYES PLLC
421 W RIVERSIDE AVENUE SUITE 500
SPOKANE, WA 99201

EXAMINER	
CHEN, QING	
ART UNIT	PAPER NUMBER
2191	

SHORTENED STATUTORY PERIOD OF RESPONSE	NOTIFICATION DATE	DELIVERY MODE
3 MONTHS	04/11/2007	ELECTRONIC

Please find below and/or attached an Office communication concerning this application or proceeding.

If NO period for reply is specified above, the maximum statutory period will apply and will expire 6 MONTHS from the mailing date of this communication.

Notice of this Office communication was sent electronically on the above-indicated "Notification Date" and has a shortened statutory period for reply of 3 MONTHS from 04/11/2007.

Notice of the Office communication was sent electronically on above-indicated "Notification Date" to the following e-mail address(es):

lhptoms@leehayes.com

Office Action Summary	Application No.	Applicant(s)	
	10/692,320	CWALINA ET AL.	
	Examiner	Art Unit	
	Qing Chen	2191	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *23 October 2003*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-59* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-59* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☒ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *23 October 2003* is/are: a)☒ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☒ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date *20040108, 20060321, 20060501*.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

1. This is the initial Office action based on the application filed on October 23, 2003.

2. **Claims 1-59** are pending.

Information Disclosure Statement

3. The information disclosure statement filed on March 21, 2006 fails to comply with 37 CFR 1.98(a)(2), which requires a legible copy of each cited foreign patent document; each non-patent literature publication or that portion which caused it to be listed; and all other information or that portion which caused it to be listed. It has been placed in the application file, but the information referred to therein has not been considered.

Specification

4. The use of trademarks, such as FIREWIRE, has been noted in this application. Trademarks should be capitalized wherever they appear (capitalize each letter OR accompany each trademark with an appropriate designation symbol, *e.g.*, ™ or ®) and be accompanied by the generic terminology (use trademarks as adjectives modifying a descriptive noun, *e.g.*, "the JAVA programming language").

Although the use of trademarks is permissible in patent applications, the proprietary nature of the marks should be respected and every effort made to prevent their use in any manner, which might adversely affect their validity as trademarks.

Claim Objections

5. **Claims 40 and 53** are objected to because of the following informalities:

- **Claim 40** contains a typographical error: the word "and" should be added after the first limitation.
- **Claim 53** contains a typographical error: the word "and" should be deleted after the first limitation.

Appropriate correction is required.

Claim Rejections - 35 USC § 112

6. The following is a quotation of the second paragraph of 35 U.S.C. 112:

> The specification shall conclude with one or more claims particularly pointing out and distinctly claiming the subject matter which the applicant regards as his invention.

7. **Claims 2-4, 6-9, 20-22, 29-31, 36-39, and 48-59** are rejected under 35 U.S.C. 112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter which applicant regards as the invention.

Claims 2-4 and 36 recite the limitation "too complex." The term "too complex" is a relative term, which renders the claims indefinite. The term "too complex" is not defined by the claims nor does the specification provide a standard for ascertaining the requisite degree and one of ordinary skill in the art would not be able to reasonably determine the scope of the invention. In the interest of compact prosecution, the Examiner subsequently does not give any patentable weight to this limitation for the purpose of further examination.

Claim 6 recites the limitation "typical." The term "typical" is a relative term, which renders the claim indefinite. The term "typical" is not defined by the claim nor does the specification provide a standard for ascertaining the requisite degree and one of ordinary skill in the art would not be able to reasonably determine the scope of the invention. In the interest of compact prosecution, the Examiner subsequently does not give any patentable weight to this limitation for the purpose of further examination.

Claims 7, 8, 20, 37, and 39 recite the limitation "significant problems." The term "significant" is a relative term, which renders the claims indefinite. The term "significant" is not defined by the claims nor does the specification provide a standard for ascertaining the requisite degree and one of ordinary skill in the art would not be able to reasonably determine the scope of the invention. In the interest of compact prosecution, the Examiner subsequently does not give any patentable weight to this limitation for the purpose of further examination.

Claim 9 depends on Claim 8 and, therefore, suffers the same deficiency as Claim 8.

Claims 21 and 22 depend on Claim 20 and, therefore, suffer the same deficiency as Claim 20.

Claim 38 depends on Claim 37 and, therefore, suffers the same deficiency as Claim 37.

Claim 26 recites the limitation "uninteresting." The term "uninteresting" is a relative term, which renders the claim indefinite. The term "uninteresting" is not defined by the claim nor

does the specification provide a standard for ascertaining the requisite degree and one of ordinary skill in the art would not be able to reasonably determine the scope of the invention. In the interest of compact prosecution, the Examiner subsequently does not give any patentable weight to this limitation for the purpose of further examination.

Claims 29 and 31 recite the limitation "relatively higher/lower." The term "relatively higher/lower" is a relative term, which renders the claims indefinite. The term "relatively higher/lower" is not defined by the claims nor does the specification provide a standard for ascertaining the requisite degree and one of ordinary skill in the art would not be able to reasonably determine the scope of the invention. In the interest of compact prosecution, the Examiner subsequently does not give any patentable weight to this limitation for the purpose of further examination.

Claim 30 depends on Claim 29 and, therefore, suffers the same deficiency as Claim 29.

Claim 30 recites the limitation "relatively higher." The term "relatively higher" is a relative term, which renders the claim indefinite. The term "relatively higher" is not defined by the claim nor does the specification provide a standard for ascertaining the requisite degree and one of ordinary skill in the art would not be able to reasonably determine the scope of the invention. In the interest of compact prosecution, the Examiner subsequently does not give any patentable weight to this limitation for the purpose of further examination.

Claim 31 recites the limitation "relatively greater." The term "relatively greater" is a relative term, which renders the claim indefinite. The term "relatively greater" is not defined by the claim nor does the specification provide a standard for ascertaining the requisite degree and one of ordinary skill in the art would not be able to reasonably determine the scope of the invention. In the interest of compact prosecution, the Examiner subsequently does not give any patentable weight to this limitation for the purpose of further examination.

Claim 39 recites the limitation "significantly different." The term "significantly" is a relative term, which renders the claim indefinite. The term "significantly" is not defined by the claim nor does the specification provide a standard for ascertaining the requisite degree and one of ordinary skill in the art would not be able to reasonably determine the scope of the invention. In the interest of compact prosecution, the Examiner subsequently does not give any patentable weight to this limitation for the purpose of further examination.

Claim 48 recites the limitations "gradual progression," "simpler situations," "increasing portion," and "complex situations." The terms "gradual," "simpler," "increasing," and "complex" are relative terms, which render the claim indefinite. The terms "gradual," "simpler," "increasing," and "complex" are not defined by the claim nor does the specification provide a standard for ascertaining the requisite degree and one of ordinary skill in the art would not be able to reasonably determine the scope of the invention. In the interest of compact prosecution, the Examiner subsequently does not give any patentable weight to these limitations for the purpose of further examination.

Claim 49 recites the limitation "undue complexity." The term "undue complexity" is a relative term, which renders the claim indefinite. The term "undue complexity" is not defined by the claim nor does the specification provide a standard for ascertaining the requisite degree and one of ordinary skill in the art would not be able to reasonably determine the scope of the invention. In the interest of compact prosecution, the Examiner subsequently does not give any patentable weight to this limitation for the purpose of further examination.

Claims 50-59 depend on Claim 49 and, therefore, suffer the same deficiency as Claim 49.

Claim 52 recites the limitation "appropriate methods, defaults, and abstractions." The term "appropriate" is a relative term, which renders the claim indefinite. The term "appropriate" is not defined by the claim nor does the specification provide a standard for ascertaining the requisite degree and one of ordinary skill in the art would not be able to reasonably determine the scope of the invention. In the interest of compact prosecution, the Examiner subsequently does not give any patentable weight to this limitation for the purpose of further examination.

Claim 53 recites the limitations "simplicity" and "sufficiently simple." The terms "simplicity" and "sufficiently simple" are relative terms, which render the claim indefinite. The terms "simplicity" and "sufficiently simple" are not defined by the claim nor does the specification provide a standard for ascertaining the requisite degree and one of ordinary skill in

the art would not be able to reasonably determine the scope of the invention. In the interest of compact prosecution, the Examiner subsequently does not give any patentable weight to these limitations for the purpose of further examination.

Claim 56 recites the limitation "ideal factoring." The term "ideal" is a relative term, which renders the claim indefinite. The term "ideal" is not defined by the claim nor does the specification provide a standard for ascertaining the requisite degree and one of ordinary skill in the art would not be able to reasonably determine the scope of the invention. In the interest of compact prosecution, the Examiner subsequently does not give any patentable weight to this limitation for the purpose of further examination.

Claim Rejections - 35 USC § 101

8. 35 U.S.C. 101 reads as follows:

> Whoever invents or discovers any new and useful process, machine, manufacture, or composition of matter, or any new and useful improvement thereof, may obtain a patent therefor, subject to the conditions and requirements of this title.

9. **Claims 1-33 and 48-59** are rejected under 35 U.S.C. 101 because the claimed invention is directed to non-statutory subject matter.

The result of **Claims 1-33 and 48** is directed to the act of "deriving," which does not appear to be a tangible result so as to constitute a practical application of the idea. The act of "deriving" is merely a thought or an abstract idea and does not appear to produce a tangible result even if the step of "deriving" does occur, since the result of that derivation is not conveyed

in the real world. The result is a derivation, which is neither used in a disclosed practical application nor made available for use in a disclosed practical application. It also does not appear that the usefulness of the derivation can be realized from the claimed steps to support a disclosed specific, substantial, and credible utility so as to produce a useful result.

Therefore, the claims do not meet the statutory requirement of 35 U.S.C. § 101, since the claims are not directed to a practical application of the § 101 judicial exception producing a result tied to the physical world.

Claims 49-59 recite the limitation of defining a plurality of factored types responsive to the deciding if additional requirements cannot be added to the at least one aggregate component without adding undue complexity to the at least one scenario. However, in the case of where the "if" condition does occur, then the claims do not produce a useful, concrete, and tangible result because the claims do not recite any step or means for the "if" condition. Therefore, the claims do not produce a practical application of the § 101 judicial exception.

The result of **Claims 49-59** is directed to the act of "deciding," which does not appear to be a tangible result so as to constitute a practical application of the idea. The act of "deciding" is merely a thought or an abstract idea and does not appear to produce a tangible result even if the step of "deciding" does occur, since the result of that decision is not conveyed in the real world.

Claims 49-59 are rejected for the same reasons set forth in the rejections of Claims 1-33 and 48.

Claim Rejections - 35 USC § 102

10. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (b) the invention was patented or described in a printed publication in this or a foreign country or in public use or on sale in this country, more than one year prior to the date of application for patent in the United States.

11. **Claims 1, 10, 11, 13-19, 23-30, 32, 33, 48-50, 52-56, 58, and 59** are rejected under 35 U.S.C. 102(b) as being anticipated by **Burger et al. (US 5,097,533)**.

As per **Claim 1**, Burger et al. disclose:

- preparing a plurality of code samples for a core scenario, each respective code sample of the plurality of code samples corresponding to a respective programming language of a plurality of programming languages *(see Column 5: 30-36, "... the invention generalizes existing entries to a plurality of applications 16 written in any of a number of predetermined languages.")*; and

- deriving the API from the core scenario responsive to the plurality of code samples *(see Column 5: 36-43, "... the function of the generic APIs 14 of the present invention is to transform parameters received in a number of different formats determined by the particular language in which the application 16 is written into forms expected by the DBM APIs 12 and operating system APIs 20 as dictated by the particular procedural language in which these APIs 12 and 20 are written.")*.

As per **Claim 10**, the rejection of **Claim 1** is incorporated; and Burger et al. further disclose:

- deriving the API to support the plurality of code samples that correspond respectively to the plurality of programming languages *(see Column 5: 36-43, "... the function of the generic APIs 14 of the present invention is to transform parameters received in a number of different formats determined by the particular language in which the application 16 is written into forms expected by the DBM APIs 12 and operating system APIs 20 as dictated by the particular procedural language in which these APIs 12 and 20 are written.").*

As per **Claim 11**, the rejection of **Claim 1** is incorporated; and Burger et al. further disclose:

- gleaning language-specific mandates from the plurality of code samples *(see Column 7: 1-10, "Services provided by the operating system 98 and accessed by system 96 are those services commonly associated with a computer's operating system. Interface entry points are provided as represented by reference numerals 60 and 62 for access to the system 96 by database applications written in languages other than the language used to write the database manager software and other than the assembly language of the machine employed in the computer system used to implement the invention.")*; and
- incorporating the language-specific mandates into the API *(see Column 6: 56-58, "Access to this system 96 is through interface entry points, i.e., function calls, indicated as the database interface 95.").*

As per **Claim 13**, the rejection of **Claim 1** is incorporated; and Burger et al. further disclose:

- gleaning commonalities from the plurality of code samples *(see Column 7: 1-10, "Services provided by the operating system 98 and accessed by system 96 are those services commonly associated with a computer's operating system. Interface entry points are provided as represented by reference numerals 60 and 62 for access to the system 96 by database applications written in languages other than the language used to write the database manager software and other than the assembly language of the machine employed in the computer system used to implement the invention.")*; and

- incorporating the commonalities into the API *(see Column 6: 56-58, "Access to this system 96 is through interface entry points, i.e., function calls, indicated as the database interface 95.").*

As per **Claim 14**, the rejection of **Claim 1** is incorporated; and Burger et al. further disclose:

- deriving the API to have an aggregate component that ties a plurality of lower-level factored types together to support the core scenario *(see Column 7: 14-26, "The memory, and more particularly a register preservation table therein, at 200 reserves a number of bytes sufficient for storing a number of processor states or entries in the table equal to the maximum number of threads allowed by the operating system 98 in an application.").*

As per **Claim 15**, Burger et al. disclose:

- selecting a core scenario for a feature area *(see Column 4: 62-67 through Column 5: 1, "These system calls provide essentially the same purpose as generic APIs 14, i.e., they provide entry points so that application programs 16 can access the operating system 18 to obtain necessary services ... ")*;

- writing at least one code sample for the core scenario *(see Column 5: 47-54, "... application 16 may be of a multi-threaded variety whereby one or more such applications may include a plurality of threads or pieces of code which may run asynchronously with respect to all remaining parts of the same or other programs or applications 16 ... ")*; and

- deriving an API for the core scenario responsive to the at least one code sample *(see Column 5: 36-43, "... the function of the generic APIs 14 of the present invention is to transform parameters received in a number of different formats determined by the particular language in which the application 16 is written into forms expected by the DBM APIs 12 and operating system APIs 20 as dictated by the particular procedural language in which these APIs 12 and 20 are written.")*.

As per **Claim 16**, the rejection of **Claim 15** is incorporated; and Burger et al. further disclose:

- selecting a plurality of core scenarios for the feature area *(see Column 5: 47-54, "... application 16 may be of a multi-threaded variety whereby one or more such applications may include a plurality of threads or pieces of code which may run asynchronously with respect to all remaining parts of the same or other programs or applications 16 ... ")*.

As per **Claim 17**, the rejection of **Claim 16** is incorporated; and Burger et al. further disclose:

- repeating the writing and the deriving for each core scenario of the plurality of core scenarios that are selected for the feature area *(see Column 4: 62-67 through Column 5: 1, "These system calls provide essentially the same purpose as generic APIs 14, i.e., they provide entry points so that application programs 16 can access the operating system 18 to obtain necessary services ... ").*

As per **Claim 18**, the rejection of **Claim 15** is incorporated; and Burger et al. further disclose:

- writing a plurality of code samples for the core scenario, each respective code sample of the plurality of code samples corresponding to a respective programming language of a plurality of programming languages *(see Column 5: 30-36, "... the invention generalizes existing entries to a plurality of applications 16 written in any of a number of predetermined languages.").*

As per **Claim 19**, the rejection of **Claim 18** is incorporated; and Burger et al. further disclose:

- deriving the API for the core scenario responsive to the plurality of code samples *(see Column 5: 36-43, "... the function of the generic APIs 14 of the present invention is to transform parameters received in a number of different formats determined by the particular language in which the application 16 is written into forms expected by the DBM APIs 12 and operating*

system APIs 20 as dictated by the particular procedural language in which these APIs 12 and 20 are written.").

As per **Claim 23**, the rejection of **Claim 15** is incorporated; and Burger et al. further disclose:

- deriving the API to support the at least one code sample written for the core scenario by producing a two-layer API that includes an aggregate component and a plurality of underlying factored types *(see Column 5: 36-43, "... the function of the generic APIs 14 of the present invention is to transform parameters received in a number of different formats determined by the particular language in which the application 16 is written into forms expected by the DBM APIs 12 and operating system APIs 20 as dictated by the particular procedural language in which these APIs 12 and 20 are written.").*

As per **Claim 24**, the rejection of **Claim 15** is incorporated; and Burger et al. further disclose:

- gleaning one or more language-specific mandates from the at least one code sample *(see Column 7: 1-10, "Services provided by the operating system 98 and accessed by system 96 are those services commonly associated with a computer's operating system. Interface entry points are provided as represented by reference numerals 60 and 62 for access to the system 96 by database applications written in languages other than the language used to write the database manager software and other than the assembly language of the machine employed in the computer system used to implement the invention.")*; and

- incorporating the one or more language-specific mandates into the API *(see Column 6: 56-58, "Access to this system 96 is through interface entry points, i.e., function calls, indicated as the database interface 95.").*

As per **Claim 25**, the rejection of **Claim 15** is incorporated; and Burger et al. further disclose:

- encapsulating a particular factored type into an aggregate component that is associated with the core scenario if all members of the particular factored type are exposed by the aggregate component *(see Column 5: 54-59, "... each time one of such threads calls an API 14 the state information of the processor in the computer executing the system of the invention must be saved in a particular manner in a state preservation table 17.").*

As per **Claim 26**, the rejection of **Claim 15** is incorporated; and Burger et al. further disclose:

- encapsulating a particular factored type into an aggregate component that is associated with the core scenario if the particular factored type is independently uninteresting to other component types *(see Column 5: 54-59, "... each time one of such threads calls an API 14 the state information of the processor in the computer executing the system of the invention must be saved in a particular manner in a state preservation table 17.").*

As per **Claim 27**, the rejection of **Claim 15** is incorporated; and Burger et al. further disclose:

- exposing a particular factored type from an aggregate component that is associated with the core scenario if at least one member of the particular factored type is not exposed by the aggregate component *(see Column 5: 54-59, "... each time one of such threads calls an API 14 the state information of the processor in the computer executing the system of the invention must be saved in a particular manner in a state preservation table 17.").*

As per **Claim 28**, the rejection of **Claim 15** is incorporated; and Burger et al. further disclose:

- exposing a particular factored type from an aggregate component that is associated with the core scenario if the particular factored type can be beneficially used independently of the aggregate component by another component type *(see Column 5: 54-59, "... each time one of such threads calls an API 14 the state information of the processor in the computer executing the system of the invention must be saved in a particular manner in a state preservation table 17.").*

As per **Claim 29**, the rejection of **Claim 15** is incorporated; and Burger et al. further disclose:

- producing a two-layer framework that includes component types targeting a relatively higher level of abstraction and component types targeting a relatively lower level of abstraction *(see Column 4: 62-67 through Column 5: 1, "These system calls provide essentially the same purpose as generic APIs 14, i.e., they provide entry points so that application programs 16 can access the operating system 18 to obtain necessary services ... ").*

As per **Claim 30**, the rejection of **Claim 29** is incorporated; and Burger et al. further disclose:

- wherein the component types targeting the relatively higher level of abstraction are directed to core scenarios *(see Column 4: 62-67 through Column 5: 1, "These system calls provide essentially the same purpose as generic APIs 14, i.e., they provide entry points so that application programs 16 can access the operating system 18 to obtain necessary services ... ").*

As per **Claim 32**, the rejection of **Claim 15** is incorporated; and Burger et al. further disclose:

- deriving the API so as to enable a developer to implement a create-set-call usage pattern for the core scenario *(see Column 5: 36-43, "... the function of the generic APIs 14 of the present invention is to transform parameters received in a number of different formats determined by the particular language in which the application 16 is written into forms expected by the DBM APIs 12 and operating system APIs 20 as dictated by the particular procedural language in which these APIs 12 and 20 are written.").*

As per **Claim 33**, the rejection of **Claim 32** is incorporated; and Burger et al. further disclose:

- producing the API with pre-selected parameters that are appropriate for the core scenario *(see Column 5: 36-43, "... the function of the generic APIs 14 of the present invention is to transform parameters received in a number of different formats determined by the particular language in which the application 16 is written into forms expected by the DBM APIs 12 and*

operating system APIs 20 as dictated by the particular procedural language in which these APIs 12 and 20 are written.").

As per **Claim 48**, Burger et al. disclose:

- writing at least one code sample for a scenario *(see Column 5: 47-54, "... application 16 may be of a multi-threaded variety whereby one or more such applications may include a plurality of threads or pieces of code which may run asynchronously with respect to all remaining parts of the same or other programs or applications 16 ...")*; and

- deriving an API for the scenario responsive to the at least one code sample, the API including (i) an aggregate component that is adapted to facilitate implementation of the scenario and (ii) a plurality of factored types that provide underlying functionality for the aggregate component, the API enabling a gradual progression from using the aggregate component in simpler situations to using an increasing portion of the plurality of factored types in increasingly complex situations *(see Column 5: 36-43, "... the function of the generic APIs 14 of the present invention is to transform parameters received in a number of different formats determined by the particular language in which the application 16 is written into forms expected by the DBM APIs 12 and operating system APIs 20 as dictated by the particular procedural language in which these APIs 12 and 20 are written."; Column 7: 14-26, "The memory, and more particularly a register preservation table therein, at 200 reserves a number of bytes sufficient for storing a number of processor states or entries in the table equal to the maximum number of threads allowed by the operating system 98 in an application.").*

As per **Claim 49**, Burger et al. disclose:

- deriving at least one aggregate component to support at least one code sample for at least one scenario *(see Column 7: 14-26, "The memory, and more particularly a register preservation table therein, at 200 reserves a number of bytes sufficient for storing a number of processor states or entries in the table equal to the maximum number of threads allowed by the operating system 98 in an application.")*;

- determining additional requirements with respect to the at least one scenario *(see Column 7: 33-37, "Upon activation by a call such as call 94, the API reads, as indicated by the arrow 70, the state of a register 52 of the processor 51 and stores it as indicated by arrow 46 in the current thread's stack 48 before it causes the processor's state to change.")*;

- deciding if the additional requirements can be added to the at least one aggregate component without adding undue complexity to the at least one scenario *(see Column 7: 33-37, "Upon activation by a call such as call 94, the API reads, as indicated by the arrow 70, the state of a register 52 of the processor 51 and stores it as indicated by arrow 46 in the current thread's stack 48 before it causes the processor's state to change.")*; and

- if not, defining a plurality of factored types responsive to the deciding *(see Column 7: 49-52, "The API 62 then proceeds to make the parameter fixes ...")*.

As per **Claim 50**, the rejection of **Claim 49** is incorporated; and Burger et al. further disclose:

- if so, refining the at least one aggregate component to incorporate the additional requirements *(see Column 7: 33-37, "Upon activation by a call such as call 94, the API reads, as*

indicated by the arrow 70, the state of a register 52 of the processor 51 and stores it as indicated by arrow 46 in the current thread's stack 48 before it causes the processor's state to change.").

As per **Claim 52**, the rejection of **Claim 49** is incorporated; and Burger et al. further disclose:

- deriving the at least one aggregate component with appropriate methods, defaults, and abstractions to support the at least one code sample for the at least one scenario *(see Column 7: 14-26, "The memory, and more particularly a register preservation table therein, at 200 reserves a number of bytes sufficient for storing a number of processor states or entries in the table equal to the maximum number of threads allowed by the operating system 98 in an application.").*

As per **Claim 53**, the rejection of **Claim 49** is incorporated; and Burger et al. further disclose:

- refining the at least one code sample according to the at least one derived aggregate component *(see Column 7: 33-37, "Upon activation by a call such as call 94, the API reads, as indicated by the arrow 70, the state of a register 52 of the processor 51 and stores it as indicated by arrow 46 in the current thread's stack 48 before it causes the processor's state to change.")*;
- evaluating the refined at least one code sample with regard to simplicity *(see Column 7: 49-52, "The API 62 then proceeds to make the parameter fixes ...")*; and

- repeating the deriving if the refined at least one code sample fails to be sufficiently simple in the evaluating *(see Column 7: 32-33, "The operating system 98 requires a separate stack for each such execution thread.")*.

As per **Claim 54**, the rejection of **Claim 49** is incorporated; and Burger et al. further disclose:

- determining additional requirements with respect to the at least one scenario, wherein the additional requirements include additional scenarios, additional usages, and additional interactions with other component types *(see Column 7: 33-37, "Upon activation by a call such as call 94, the API reads, as indicated by the arrow 70, the state of a register 52 of the processor 51 and stores it as indicated by arrow 46 in the current thread's stack 48 before it causes the processor's state to change.")*.

As per **Claim 55**, the rejection of **Claim 49** is incorporated; and Burger et al. further disclose:

- considering whether adding the additional requirements to the at least one aggregate component hinders a create-set-call usage pattern *(see Column 7: 33-37, "Upon activation by a call such as call 94, the API reads, as indicated by the arrow 70, the state of a register 52 of the processor 51 and stores it as indicated by arrow 46 in the current thread's stack 48 before it causes the processor's state to change.")*.

As per **Claim 56**, the rejection of **Claim 49** is incorporated; and Burger et al. further disclose:

- defining the plurality of factored types responsive to the deciding with an ideal factoring of a full set of functionality *(see Column 5: 36-43, "... the function of the generic APIs 14 of the present invention is to transform parameters received in a number of different formats determined by the particular language in which the application 16 is written into forms expected by the DBM APIs 12 and operating system APIs 20 as dictated by the particular procedural language in which these APIs 12 and 20 are written.").*

As per **Claim 58**, the rejection of **Claim 49** is incorporated; and Burger et al. further disclose:

- determining whether the at least one aggregate component is to encapsulate or expose the functionality of each factored type of the plurality of factored types *(see Column 5: 54-59, "... each time one of such threads calls an API 14 the state information of the processor in the computer executing the system of the invention must be saved in a particular manner in a state preservation table 17.").*

As per **Claim 59**, the rejection of **Claim 49** is incorporated; and Burger et al. further disclose:

- refining the plurality of factored types to support the at least one aggregate component and the additional requirements *(see Column 7: 33-37, "Upon activation by a call such as call 94, the API reads, as indicated by the arrow 70, the state of a register 52 of the*

processor 51 and stores it as indicated by arrow 46 in the current thread's stack 48 before it causes the processor's state to change.").

Claim Rejections - 35 USC § 103

12. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

13. **Claims 2-4** are rejected under 35 U.S.C. 103(a) as being unpatentable over **Burger et al. (US 5,097,533)** in view of **Hayes (US 6,006,279)**.

As per **Claim 2**, the rejection of **Claim 1** is incorporated; however, Burger et al. do not disclose:

- determining, by an API designer, if the derived API is too complex.

Hayes discloses:

- determining, by an API designer, if the derived API is too complex *(see Column 1: 35-37, "... the Photoshop native plug-in API is defined by Adobe developers." and 40-42, "Plug-in API's have become more complex as the functionality provided to client applications through plug-in modules has become more sophisticated.").*

Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate the teaching of Hayes into the teaching of Burger et al. to

include determining, by an API designer, if the derived API is too complex. The modification would be obvious because one of ordinary skill in the art would be motivated to provide a simple API *(see Hayes – Column 1: 60-63).*

As per **Claim 3**, the rejection of **Claim 2** is incorporated; however, Burger et al. do not disclose:

- if the derived API is determined to be too complex, then refining, by the API designer, the derived API to produce a refined API.

Hayes discloses:

- if the derived API is determined to be too complex, then refining, by the API designer, the derived API to produce a refined API *(see Column 1: 66-67 through Column 2: 1-3, "It is therefore an object of the invention to provide a host framework which allows a client application to utilize a single simple interface in order to access plug-in modules conforming to any of several past and future native plug-in API's.").*

Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate the teaching of Hayes into the teaching of Burger et al. to include if the derived API is determined to be too complex, then refining, by the API designer, the derived API to produce a refined API. The modification would be obvious because one of ordinary skill in the art would be motivated to provide a simple API *(see Hayes – Column 1: 60-63).*

As per **Claim 4**, the rejection of **Claim 3** is incorporated; however, Burger et al. do not disclose:

- determining, by the API designer, if the refined API is too complex.

Hayes discloses:

- determining, by the API designer, if the refined API is too complex *(see Column 1: 35-37, "... the Photoshop native plug-in API is defined by Adobe developers." and 40-42, "Plug-in API's have become more complex as the functionality provided to client applications through plug-in modules has become more sophisticated.").*

Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate the teaching of Hayes into the teaching of Burger et al. to include determining, by the API designer, if the refined API is too complex. The modification would be obvious because one of ordinary skill in the art would be motivated to provide a simple API *(see Hayes – Column 1: 60-63).*

14. **Claims 5-9, 12, 20-22, 31, 34, 35, 37-45, and 47** are rejected under 35 U.S.C. 103(a) as being unpatentable over **Burger et al. (US 5,097,533)** in view of **Corrie, Jr. et al. (US 5,495,571)**.

As per **Claim 5**, the rejection of **Claim 1** is incorporated; however, Burger et al. do not disclose:

- performing one or more usability studies on the API utilizing a plurality of developers.

Corrie, Jr. et al. disclose:

- performing one or more usability studies on the API utilizing a plurality of developers *(see Column 2: 4-8, "When testing all of the functions in the application programming interface to Microsoft Windows, for example, programmers must write modules of test code for approximately 1,000 functions, each function having on an average of 3-4 parameters.").*

Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate the teaching of Corrie, Jr. et al. into the teaching of Burger et al. to include performing one or more usability studies on the API utilizing a plurality of developers. The modification would be obvious because one of ordinary skill in the art would be motivated to save money by preventing expensive product updates to correct errors, and the high product quality ensured by the thorough testing would enhance a software product's reputation and marketability *(see Corrie, Jr. et al. – Column 2: 22-26).*

As per **Claim 6**, the rejection of **Claim 5** is incorporated; however, Burger et al. do not disclose:

- performing the one or more usability studies on the API utilizing the plurality of developers wherein the plurality of developers are typical for the plurality of programming languages.

Corrie, Jr. et al. disclose:

- performing the one or more usability studies on the API utilizing the plurality of developers wherein the plurality of developers are typical for the plurality of programming languages *(see Column 2: 4-8, "When testing all of the functions in the application programming*

interface to Microsoft Windows, for example, programmers must write modules of test code for approximately 1,000 functions, each function having on an average of 3-4 parameters.").

Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate the teaching of Corrie, Jr. et al. into the teaching of Burger et al. to include performing the one or more usability studies on the API utilizing the plurality of developers wherein the plurality of developers are typical for the plurality of programming languages. The modification would be obvious because one of ordinary skill in the art would be motivated to save money by preventing expensive product updates to correct errors, and the high product quality ensured by the thorough testing would enhance a software product's reputation and marketability *(see Corrie, Jr. et al. – Column 2: 22-26).*

As per **Claim 7**, the rejection of **Claim 5** is incorporated; however, Burger et al. do not disclose:

- ascertaining whether the plurality of developers are able to use the API without significant problems.

Corrie, Jr. et al. disclose:

- ascertaining whether the plurality of developers are able to use the API without significant problems *(see Column 1: 52-57, "Programmers perform parametric testing of functions to test how functions react to valid and invalid parameters.").*

Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate the teaching of Corrie, Jr. et al. into the teaching of Burger et al. to include ascertaining whether the plurality of developers are able to use the API without

significant problems. The modification would be obvious because one of ordinary skill in the art would be motivated to save money by preventing expensive product updates to correct errors, and the high product quality ensured by the thorough testing would enhance a software product's reputation and marketability *(see Corrie, Jr. et al. – Column 2: 22-26).*

As per **Claim 8**, the rejection of **Claim 7** is incorporated; however, Burger et al. do not disclose:

- if the plurality of developers are not ascertained to be able to use the API without significant problems, then revising the API.

Corrie, Jr. et al. disclose:

- if the plurality of developers are not ascertained to be able to use the API without significant problems, then revising the API *(see Column 1: 52-57, "To perform parametric testing of a function in prior art systems, programmers write modules of test code to pass valid and invalid parameters to the function, and then observe the behavior of the function."; Column 2: 52-55, "The test utility coordinates the parametric testing of each function in the functional programming interface. The test utility reads the script file and generates a plurality of unique test cases.").*

Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate the teaching of Corrie, Jr. et al. into the teaching of Burger et al. to include if the plurality of developers are not ascertained to be able to use the API without significant problems, then revising the API. The modification would be obvious because one of ordinary skill in the art would be motivated to save money by preventing expensive product

updates to correct errors, and the high product quality ensured by the thorough testing would enhance a software product's reputation and marketability *(see Corrie, Jr. et al. – Column 2: 22-26).*

As per **Claim 9**, the rejection of **Claim 8** is incorporated; however, Burger et al. do not disclose:

- revising the API based on at least one lesson from the one or more usability studies.

Corrie, Jr. et al. disclose:

- revising the API based on at least one lesson from the one or more usability studies *(see Column 2: 52-55, "The test utility coordinates the parametric testing of each function in the functional programming interface. The test utility reads the script file and generates a plurality of unique test cases.").*

Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate the teaching of Corrie, Jr. et al. into the teaching of Burger et al. to include revising the API based on at least one lesson from the one or more usability studies. The modification would be obvious because one of ordinary skill in the art would be motivated to save money by preventing expensive product updates to correct errors, and the high product quality ensured by the thorough testing would enhance a software product's reputation and marketability *(see Corrie, Jr. et al. – Column 2: 22-26).*

As per **Claim 12**, the rejection of **Claim 1** is incorporated; however, Burger et al. do not disclose:

- gleaning language-inspired developer expectations from the plurality of code samples; and

- incorporating the language-inspired developer expectations into the API.

Corrie, Jr. et al. disclose:

- gleaning language-inspired developer expectations from the plurality of code samples *(see Column 1: 52-57, "To perform parametric testing of a function in prior art systems, programmers write modules of test code to pass valid and invalid parameters to the function, and then observe the behavior of the function.")*; and

- incorporating the language-inspired developer expectations into the API *(see Column 2: 52-55, "The test utility coordinates the parametric testing of each function in the functional programming interface. The test utility reads the script file and generates a plurality of unique test cases.")*.

Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate the teaching of Corrie, Jr. et al. into the teaching of Burger et al. to include gleaning language-inspired developer expectations from the plurality of code samples; and incorporating the language-inspired developer expectations into the API. The modification would be obvious because one of ordinary skill in the art would be motivated to save money by preventing expensive product updates to correct errors, and the high product quality ensured by the thorough testing would enhance a software product's reputation and marketability *(see Corrie, Jr. et al. – Column 2: 22-26)*.

As per **Claim 20**, the rejection of **Claim 15** is incorporated; however, Burger et al. do not disclose:

- performing one or more usability studies on the API utilizing a plurality of developers;

- ascertaining whether the plurality of developers are able to use the API without significant problems; and

- if the plurality of developers are not ascertained to be able to use the API without significant problems, then revising the API.

Corrie, Jr. et al. disclose:

- performing one or more usability studies on the API utilizing a plurality of developers *(see Column 2: 4-8, "When testing all of the functions in the application programming interface to Microsoft Windows, for example, programmers must write modules of test code for approximately 1,000 functions, each function having on an average of 3-4 parameters.")*;

- ascertaining whether the plurality of developers are able to use the API without significant problems *(see Column 1: 52-57, "Programmers perform parametric testing of functions to test how functions react to valid and invalid parameters.")*; and

- if the plurality of developers are not ascertained to be able to use the API without significant problems, then revising the API *(see Column 1: 52-57, "To perform parametric testing of a function in prior art systems, programmers write modules of test code to pass valid and invalid parameters to the function, and then observe the behavior of the function."; Column 2: 52-55, "The test utility coordinates the parametric testing of each function in the functional*

programming interface. The test utility reads the script file and generates a plurality of unique test cases.").

Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate the teaching of Corrie, Jr. et al. into the teaching of Burger et al. to include performing one or more usability studies on the API utilizing a plurality of developers; ascertaining whether the plurality of developers are able to use the API without significant problems; and if the plurality of developers are not ascertained to be able to use the API without significant problems, then revising the API. The modification would be obvious because one of ordinary skill in the art would be motivated to save money by preventing expensive product updates to correct errors, and the high product quality ensured by the thorough testing would enhance a software product's reputation and marketability *(see Corrie, Jr. et al. – Column 2: 22-26).*

As per **Claim 21**, the rejection of **Claim 20** is incorporated; however, Burger et al. do not disclose:

- revising the API based on at least one lesson from the one or more usability studies to produce a revised API.

Corrie, Jr. et al. disclose:

- revising the API based on at least one lesson from the one or more usability studies to produce a revised API *(see Column 2: 52-55, "The test utility coordinates the parametric testing of each function in the functional programming interface. The test utility reads the script file and generates a plurality of unique test cases.").*

Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate the teaching of Corrie, Jr. et al. into the teaching of Burger et al. to include revising the API based on at least one lesson from the one or more usability studies to produce a revised API. The modification would be obvious because one of ordinary skill in the art would be motivated to save money by preventing expensive product updates to correct errors, and the high product quality ensured by the thorough testing would enhance a software product's reputation and marketability *(see Corrie, Jr. et al. – Column 2: 22-26).*

As per **Claim 22**, the rejection of **Claim 21** is incorporated; however, Burger et al. do not disclose:

- repeating the performing and the ascertaining with respect to the revised API.

Corrie, Jr. et al. disclose:

- repeating the performing and the ascertaining with respect to the revised API *(see Column 1: 52-57, "Programmers perform parametric testing of functions to test how functions react to valid and invalid parameters."; Column 2: 4-8, "When testing all of the functions in the application programming interface to Microsoft Windows, for example, programmers must write modules of test code for approximately 1,000 functions, each function having on an average of 3-4 parameters.").*

Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate the teaching of Corrie, Jr. et al. into the teaching of Burger et al. to include repeating the performing and the ascertaining with respect to the revised API. The modification would be obvious because one of ordinary skill in the art would be motivated to

save money by preventing expensive product updates to correct errors, and the high product quality ensured by the thorough testing would enhance a software product's reputation and marketability *(see Corrie, Jr. et al. – Column 2: 22-26).*

As per **Claim 31**, the rejection of **Claim 29** is incorporated; however, Burger et al. do not disclose:

- wherein the component types targeting the relatively lower level of abstraction provide a relatively greater amount of control to developers as compared to the component types targeting the relatively higher level of abstraction.

Corrie, Jr. et al. disclose:

- wherein the component types targeting the relatively lower level of abstraction provide a relatively greater amount of control to developers as compared to the component types targeting the relatively higher level of abstraction *(see Column 1: 52-57, "To perform parametric testing of a function in prior art systems, programmers write modules of test code to pass valid and invalid parameters to the function, and then observe the behavior of the function.").*

Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate the teaching of Corrie, Jr. et al. into the teaching of Burger et al. to include wherein the component types targeting the relatively lower level of abstraction provide a relatively greater amount of control to developers as compared to the component types targeting the relatively higher level of abstraction. The modification would be obvious because one of ordinary skill in the art would be motivated to save money by preventing expensive product updates to correct errors, and the high product quality ensured by the thorough testing

would enhance a software product's reputation and marketability *(see Corrie, Jr. et al. – Column 2: 22-26).*

As per **Claim 34**, Burger et al. disclose:

- deriving an API for a scenario responsive to at least one code sample written with regard to the scenario *(see Column 5: 36-43, "... the function of the generic APIs 14 of the present invention is to transform parameters received in a number of different formats determined by the particular language in which the application 16 is written into forms expected by the DBM APIs 12 and operating system APIs 20 as dictated by the particular procedural language in which these APIs 12 and 20 are written.").*

However, Burger et al. do not disclose:

- performing one or more usability studies on the API utilizing a plurality of developers; and

- revising the API based on the one or more usability studies.

Corrie, Jr. et al. disclose:

- performing one or more usability studies on the API utilizing a plurality of developers *(see Column 2: 4-8, "When testing all of the functions in the application programming interface to Microsoft Windows, for example, programmers must write modules of test code for approximately 1,000 functions, each function having on an average of 3-4 parameters.")*; and

- revising the API based on the one or more usability studies *(see Column 2: 52-55, "The test utility coordinates the parametric testing of each function in the functional*

programming interface. The test utility reads the script file and generates a plurality of unique test cases.").

Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate the teaching of Corrie, Jr. et al. into the teaching of Burger et al. to include performing one or more usability studies on the API utilizing a plurality of developers; and revising the API based on the one or more usability studies. The modification would be obvious because one of ordinary skill in the art would be motivated to save money by preventing expensive product updates to correct errors, and the high product quality ensured by the thorough testing would enhance a software product's reputation and marketability *(see Corrie, Jr. et al. – Column 2: 22-26).*

As per **Claim 35**, the rejection of **Claim 34** is incorporated; and Burger et al. further disclose:

- writing a plurality of code samples with regard to the scenario, each respective code sample of the plurality of code samples corresponding to a respective programming language of a plurality of programming languages; wherein the deriving comprises: deriving the API for the scenario responsive to the plurality of code samples *(see Column 5: 36-43, "... the function of the generic APIs 14 of the present invention is to transform parameters received in a number of different formats determined by the particular language in which the application 16 is written into forms expected by the DBM APIs 12 and operating system APIs 20 as dictated by the particular procedural language in which these APIs 12 and 20 are written.").*

As per **Claim 37**, the rejection of **Claim 34** is incorporated; however, Burger et al. do not disclose:

- ascertaining whether the plurality of developers are able to use the API without significant problems; and

- when the plurality of developers are not ascertained to be able to use the API without significant problems, then implementing the revising; wherein the revising comprises: revising the API based on at least one lesson from the one or more usability studies to produce a revised API.

Corrie, Jr. et al. disclose:

- ascertaining whether the plurality of developers are able to use the API without significant problems *(see Column 1: 52-57, "Programmers perform parametric testing of functions to test how functions react to valid and invalid parameters.")*; and

- when the plurality of developers are not ascertained to be able to use the API without significant problems, then implementing the revising; wherein the revising comprises: revising the API based on at least one lesson from the one or more usability studies to produce a revised API *(see Column 1: 52-57, "To perform parametric testing of a function in prior art systems, programmers write modules of test code to pass valid and invalid parameters to the function, and then observe the behavior of the function."; Column 2: 52-55, "The test utility coordinates the parametric testing of each function in the functional programming interface. The test utility reads the script file and generates a plurality of unique test cases.")*.

Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate the teaching of Corrie, Jr. et al. into the teaching of Burger et

al. to include ascertaining whether the plurality of developers are able to use the API without significant problems; and when the plurality of developers are not ascertained to be able to use the API without significant problems, then implementing the revising; wherein the revising comprises: revising the API based on at least one lesson from the one or more usability studies to produce a revised API. The modification would be obvious because one of ordinary skill in the art would be motivated to save money by preventing expensive product updates to correct errors, and the high product quality ensured by the thorough testing would enhance a software product's reputation and marketability *(see Corrie, Jr. et al. – Column 2: 22-26).*

As per **Claim 38**, the rejection of **Claim 37** is incorporated; however, Burger et al. do not disclose:

- repeating at least the performing and the ascertaining with respect to the revised API.

Corrie, Jr. et al. disclose:

- repeating at least the performing and the ascertaining with respect to the revised API

(see Column 1: 52-57, "Programmers perform parametric testing of functions to test how functions react to valid and invalid parameters."; Column 2: 4-8, "When testing all of the functions in the application programming interface to Microsoft Windows, for example, programmers must write modules of test code for approximately 1,000 functions, each function having on an average of 3-4 parameters.").

Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate the teaching of Corrie, Jr. et al. into the teaching of Burger et al. to include repeating at least the performing and the ascertaining with respect to the revised

API. The modification would be obvious because one of ordinary skill in the art would be motivated to save money by preventing expensive product updates to correct errors, and the high product quality ensured by the thorough testing would enhance a software product's reputation and marketability *(see Corrie, Jr. et al. – Column 2: 22-26).*

As per **Claim 39**, the rejection of **Claim 37** is incorporated; however, Burger et al. do not disclose:

- ascertaining whether the plurality of developers are able to use the API without significant problems with regard to a desired level of usability for at least one targeted developer group, wherein the desired level of usability includes considerations with respect to (i) frequent and/or extensive reference to detailed API documentation by the plurality of developers, (ii) a failure of a majority of the plurality of developers to implement the scenario, and (iii) whether the plurality of developers take an approach that is significantly different from what is expected by an API designer.

Corrie, Jr. et al. disclose:

- ascertaining whether the plurality of developers are able to use the API without significant problems with regard to a desired level of usability for at least one targeted developer group, wherein the desired level of usability includes considerations with respect to (i) frequent and/or extensive reference to detailed API documentation by the plurality of developers, (ii) a failure of a majority of the plurality of developers to implement the scenario, and (iii) whether the plurality of developers take an approach that is significantly different from what is expected by an API designer *(see Column 1: 52-57, "Programmers perform parametric testing of*

functions to test how functions react to valid and invalid parameters."; Column 2: 33-45, "A script file, a function library, and a test utility are all used in this testing process. The script file contains prototype information for each function to be tested and a case specification for each parameter of the function.").

Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate the teaching of Corrie, Jr. et al. into the teaching of Burger et al. to include ascertaining whether the plurality of developers are able to use the API without significant problems with regard to a desired level of usability for at least one targeted developer group, wherein the desired level of usability includes considerations with respect to (i) frequent and/or extensive reference to detailed API documentation by the plurality of developers, (ii) a failure of a majority of the plurality of developers to implement the scenario, and (iii) whether the plurality of developers take an approach that is significantly different from what is expected by an API designer. The modification would be obvious because one of ordinary skill in the art would be motivated to save money by preventing expensive product updates to correct errors, and the high product quality ensured by the thorough testing would enhance a software product's reputation and marketability *(see Corrie, Jr. et al. – Column 2: 22-26).*

As per **Claim 40**, the rejection of **Claim 34** is incorporated; and Burger et al. further disclose:

- selecting a plurality of core scenarios for a feature area *(see Column 4: 62-67 through Column 5: 1, "These system calls provide essentially the same purpose as generic APIs 14, i.e.,*

they provide entry points so that application programs 16 can access the operating system 18 to obtain necessary services ... ").

However, Burger et al. do not disclose:

- repeating the deriving, the performing, and the revising for each core scenario of the plurality of core scenarios.

Corrie, Jr. et al. disclose:

- repeating the deriving, the performing, and the revising for each core scenario of the plurality of core scenarios *(see Column 2: 4-8, "When testing all of the functions in the application programming interface to Microsoft Windows, for example, programmers must write modules of test code for approximately 1,000 functions, each function having on an average of 3-4 parameters." and 52-55, "The test utility coordinates the parametric testing of each function in the functional programming interface. The test utility reads the script file and generates a plurality of unique test cases.").*

Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate the teaching of Corrie, Jr. et al. into the teaching of Burger et al. to include repeating the deriving, the performing, and the revising for each core scenario of the plurality of core scenarios. The modification would be obvious because one of ordinary skill in the art would be motivated to save money by preventing expensive product updates to correct errors, and the high product quality ensured by the thorough testing would enhance a software product's reputation and marketability *(see Corrie, Jr. et al. – Column 2: 22-26).*

As per **Claim 41**, the rejection of **Claim 40** is incorporated; and Burger et al. further disclose:

- producing an aggregate component for each core scenario of the plurality of core scenarios *(see Column 7: 14-26, "The memory, and more particularly a register preservation table therein, at 200 reserves a number of bytes sufficient for storing a number of processor states or entries in the table equal to the maximum number of threads allowed by the operating system 98 in an application.").*

As per **Claim 42**, the rejection of **Claim 34** is incorporated; and Burger et al. further disclose:

- producing an aggregate component that has a respective relationship with each respective factored type of a plurality of factored types *(see Column 7: 14-26, "The memory, and more particularly a register preservation table therein, at 200 reserves a number of bytes sufficient for storing a number of processor states or entries in the table equal to the maximum number of threads allowed by the operating system 98 in an application.").*

As per **Claim 43**, the rejection of **Claim 42** is incorporated; and Burger et al. further disclose:

- producing the aggregate component to support the scenario for which the at least one code sample is written *(see Column 7: 14-26, "The memory, and more particularly a register preservation table therein, at 200 reserves a number of bytes sufficient for storing a number of*

processor states or entries in the table equal to the maximum number of threads allowed by the operating system 98 in an application.").

As per **Claim 44**, the rejection of **Claim 42** is incorporated; and Burger et al. further disclose:

- producing the aggregate component to have an exposed relationship with at least one factored type of the plurality of factored types and an encapsulated relationship with at least one other factored type of the plurality of factored types *(see Column 7: 14-26, "The memory, and more particularly a register preservation table therein, at 200 reserves a number of bytes sufficient for storing a number of processor states or entries in the table equal to the maximum number of threads allowed by the operating system 98 in an application.").*

As per **Claim 45**, the rejection of **Claim 44** is incorporated; and Burger et al. further disclose:

- wherein the at least one other factored type of the plurality of factored types that has the encapsulated relationship with the aggregate component can be handed off by the aggregate component for direct interaction with another component type *(see Column 5: 54-59, "... each time one of such threads calls an API 14 the state information of the processor in the computer executing the system of the invention must be saved in a particular manner in a state preservation table 17.").*

As per **Claim 47**, Burger et al. disclose:

- preparing a plurality of code samples for a core scenario, each respective code sample of the plurality of code samples corresponding to a respective programming language of a plurality of programming languages *(see Column 5: 30-36, "... the invention generalizes existing entries to a plurality of applications 16 written in any of a number of predetermined languages.")*; and

- deriving the API for the core scenario responsive to the plurality of code samples *(see Column 5: 36-43, "... the function of the generic APIs 14 of the present invention is to transform parameters received in a number of different formats determined by the particular language in which the application 16 is written into forms expected by the DBM APIs 12 and operating system APIs 20 as dictated by the particular procedural language in which these APIs 12 and 20 are written.")*.

However, Burger et al. do not disclose:

- performing one or more usability studies on the API utilizing a plurality of developers; and

- revising the API based on the one or more usability studies.

Corrie, Jr. et al. disclose:

- performing one or more usability studies on the API utilizing a plurality of developers *(see Column 2: 4-8, "When testing all of the functions in the application programming interface to Microsoft Windows, for example, programmers must write modules of test code for approximately 1,000 functions, each function having on an average of 3-4 parameters.")*; and

- revising the API based on the one or more usability studies *(see Column 2: 52-55, "The test utility coordinates the parametric testing of each function in the functional*

programming interface. The test utility reads the script file and generates a plurality of unique test cases.").

Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate the teaching of Corrie, Jr. et al. into the teaching of Burger et al. to include performing one or more usability studies on the API utilizing a plurality of developers; and revising the API based on the one or more usability studies. The modification would be obvious because one of ordinary skill in the art would be motivated to save money by preventing expensive product updates to correct errors, and the high product quality ensured by the thorough testing would enhance a software product's reputation and marketability *(see Corrie, Jr. et al. – Column 2: 22-26).*

15. **Claim 36** is rejected under 35 U.S.C. 103(a) as being unpatentable over **Burger et al. (US 5,097,533)** in view of **Corrie, Jr. et al. (US 5,495,571)** as applied to Claim 34 above, and further in view of **Hayes (US 6,006,279)**.

As per **Claim 36**, the rejection of **Claim 34** is incorporated; however, Burger et al. and Corrie, Jr. et al. do not disclose:

- determining, by an API designer, if the derived API is too complex; if the derived API is determined to be too complex, then refining, by the API designer, the derived API to produce a refined API; and determining, by the API designer, if the refined API is too complex.

Hayes discloses:

- determining, by an API designer, if the derived API is too complex; if the derived API is determined to be too complex, then refining, by the API designer, the derived API to produce a refined API; and determining, by the API designer, if the refined API is too complex *(see Column 1: 40-42, "Plug-in API's have become more complex as the functionality provided to client applications through plug-in modules has become more sophisticated." and 66-67 through Column 2: 1-3, "It is therefore an object of the invention to provide a host framework which allows a client application to utilize a single simple interface in order to access plug-in modules conforming to any of several past and future native plug-in API's.").*

Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate the teaching of Hayes into the teaching of Burger et al. to include determining, by an API designer, if the derived API is too complex; if the derived API is determined to be too complex, then refining, by the API designer, the derived API to produce a refined API; and determining, by the API designer, if the refined API is too complex. The modification would be obvious because one of ordinary skill in the art would be motivated to provide a simple API *(see Hayes – Column 1: 60-63).*

16. **Claims 46, 51, and 57** are rejected under 35 U.S.C. 103(a) as being unpatentable over **Burger et al. (US 5,097,533)**.

As per **Claim 46**, the rejection of **Claim 42** is incorporated; however, Burger et al. do not disclose:

- wherein the plurality of factored types are designed using an object-oriented methodology.

Official Notice is taken that it is old and well known within the computing art to utilize object-oriented methodology. Object-oriented programming has been used extensively to implement applications and computer programs. Many popular languages support object-oriented features, such as C++ and Java™. Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to include wherein the plurality of factored types are designed using an object-oriented methodology. The modification would be obvious because one of ordinary skill in the art would be motivated to incorporate modularity into software implementation.

As per **Claim 51**, the rejection of **Claim 49** is incorporated; and Burger et al. further disclose:

- selecting a plurality of core scenarios for a feature area, the plurality of core scenarios including the at least one scenario *(see Column 5: 47-54, "... application 16 may be of a multi-threaded variety whereby one or more such applications may include a plurality of threads or pieces of code which may run asynchronously with respect to all remaining parts of the same or other programs or applications 16 ...")*; and

- writing a plurality of code samples for the plurality of core scenarios, the plurality of code samples including the at least one code sample; wherein the deriving comprises: deriving a plurality of aggregate components, which include the at least one aggregate component, to support the plurality of code samples for the plurality of core scenarios *(see Column 5: 30-36,*

"... the invention generalizes existing entries to a plurality of applications 16 written in any of a number of predetermined languages.").

However, Burger et al. do not disclose:

- showing preferred lines of code.

Official Notice is taken that it is old and well known within the computing art to show preferred lines of code. Software editing programs commonly show preferred lines of code as a feature of the user interface. Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to show preferred lines of code. The modification would be obvious because one of ordinary skill in the art would be motivated to enhance usability.

As per **Claim 57**, the rejection of **Claim 49** is incorporated; however, Burger et al. do not disclose:

- defining the plurality of factored types responsive to the deciding using one or more object-oriented methodologies.

Official Notice is taken that it is old and well known within the computing art to utilize object-oriented methodology. Object-oriented programming has been used extensively to implement applications and computer programs. Many popular languages support object-oriented features, such as C++ and Java™. Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to include defining the plurality of factored types responsive to the deciding using one or more object-oriented methodologies. The modification would be obvious because one of ordinary skill in the art would be motivated to incorporate modularity into software implementation.

Conclusion

17. The prior art made of record and not relied upon is considered pertinent to applicant's disclosure.

Any inquiry concerning this communication or earlier communications from the Examiner should be directed to Qing Chen whose telephone number is 571-270-1071. The Examiner can normally be reached on Monday through Thursday from 7:30 AM to 4:00 PM. The Examiner can also be reached on alternate Fridays.

If attempts to reach the Examiner by telephone are unsuccessful, the Examiner's supervisor, Wei Zhen, can be reached on 571-272-3708. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Any inquiry of a general nature or relating to the status of this application or proceeding should be directed to the TC 2100 Group receptionist whose telephone number is 571-272-2100.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR

system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

QC / QC
March 27, 2007

WEI ZHEN
SUPERVISORY PATENT EXAMINER